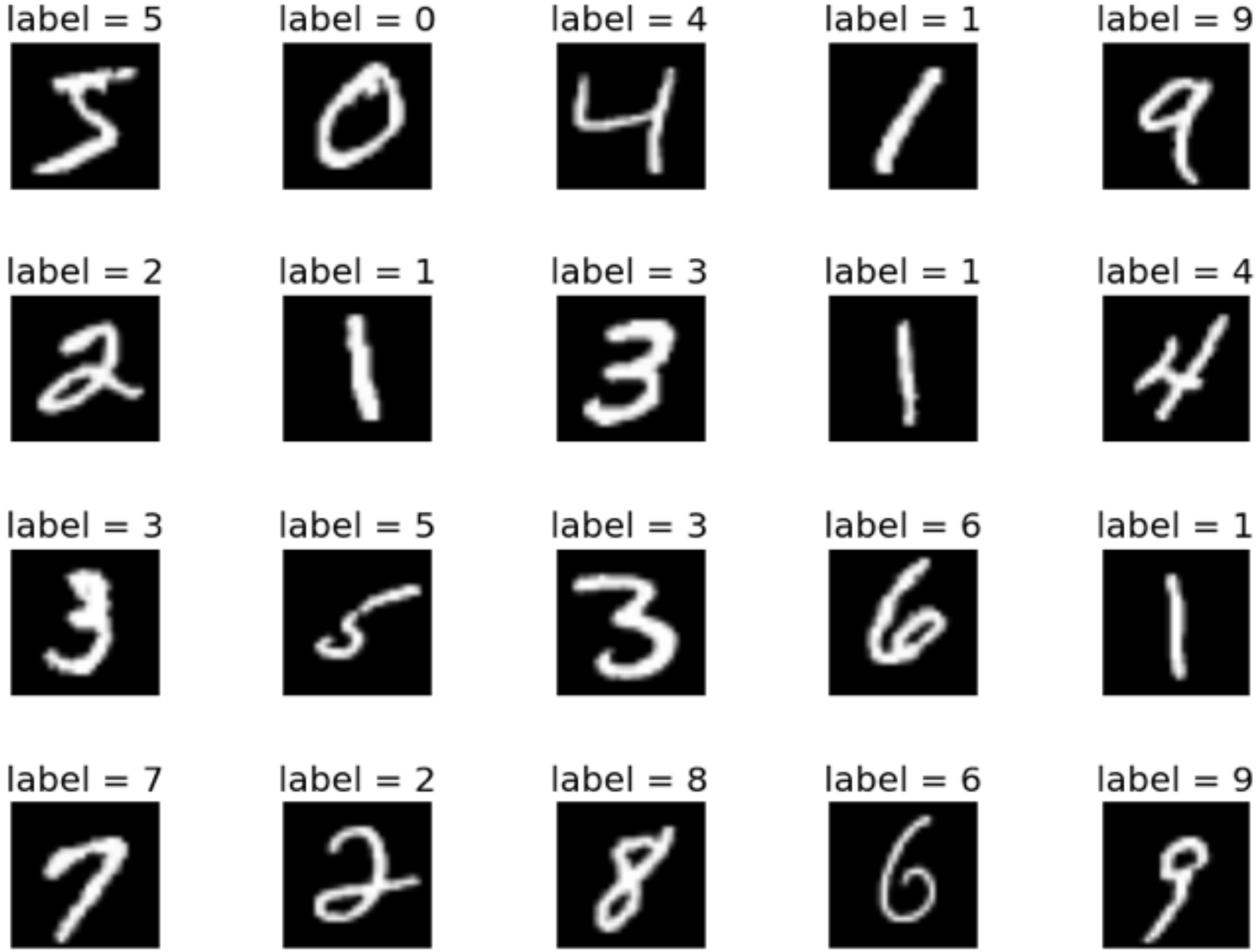
Deep Learning with MXNet Gluon

GLUON

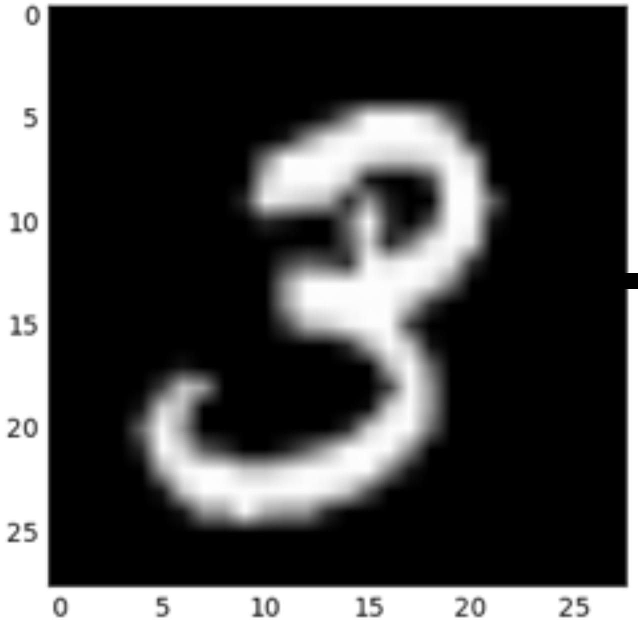
Hongyuan Mei
Center for Language and Speech Processing
Department of Computer Science, Johns Hopkins University

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

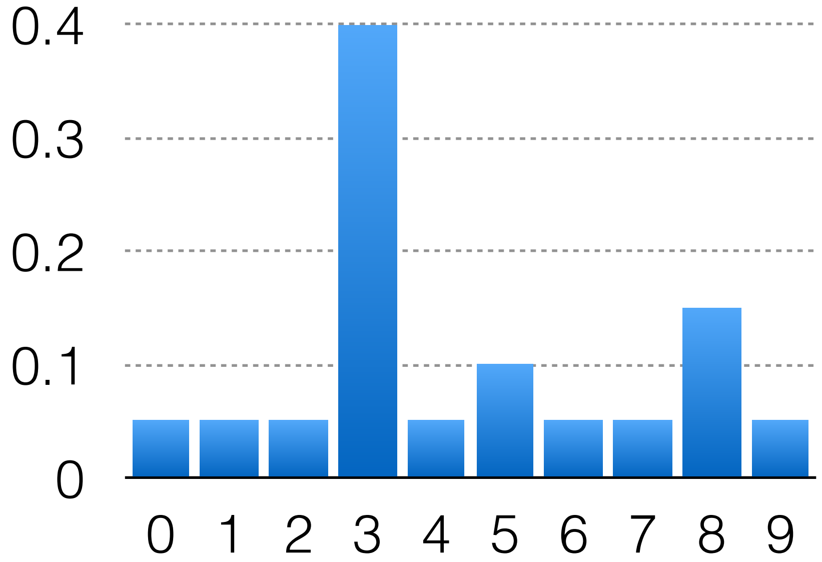
Supervised Learning



Supervised Learning

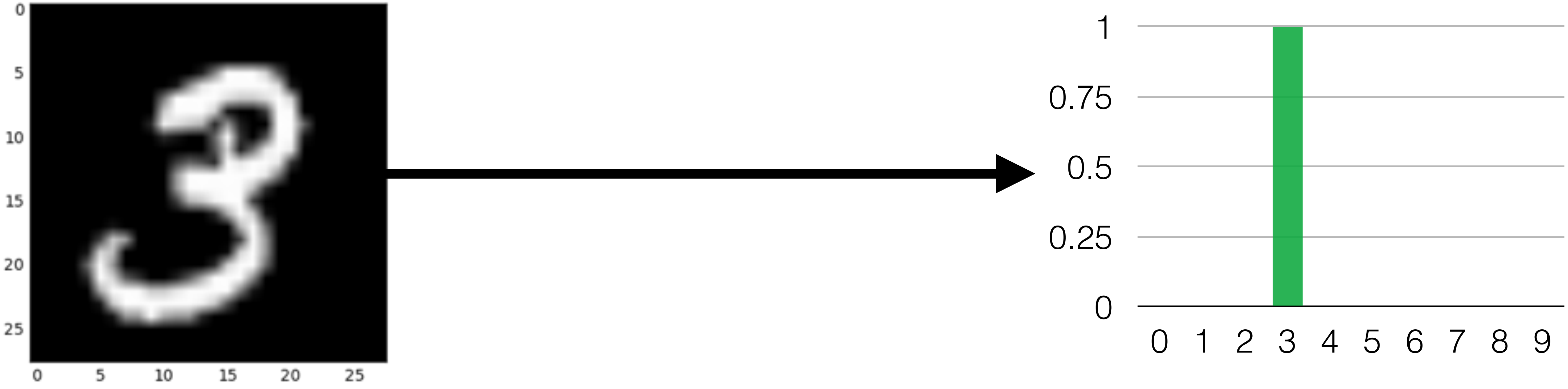


$$\mathcal{M}$$



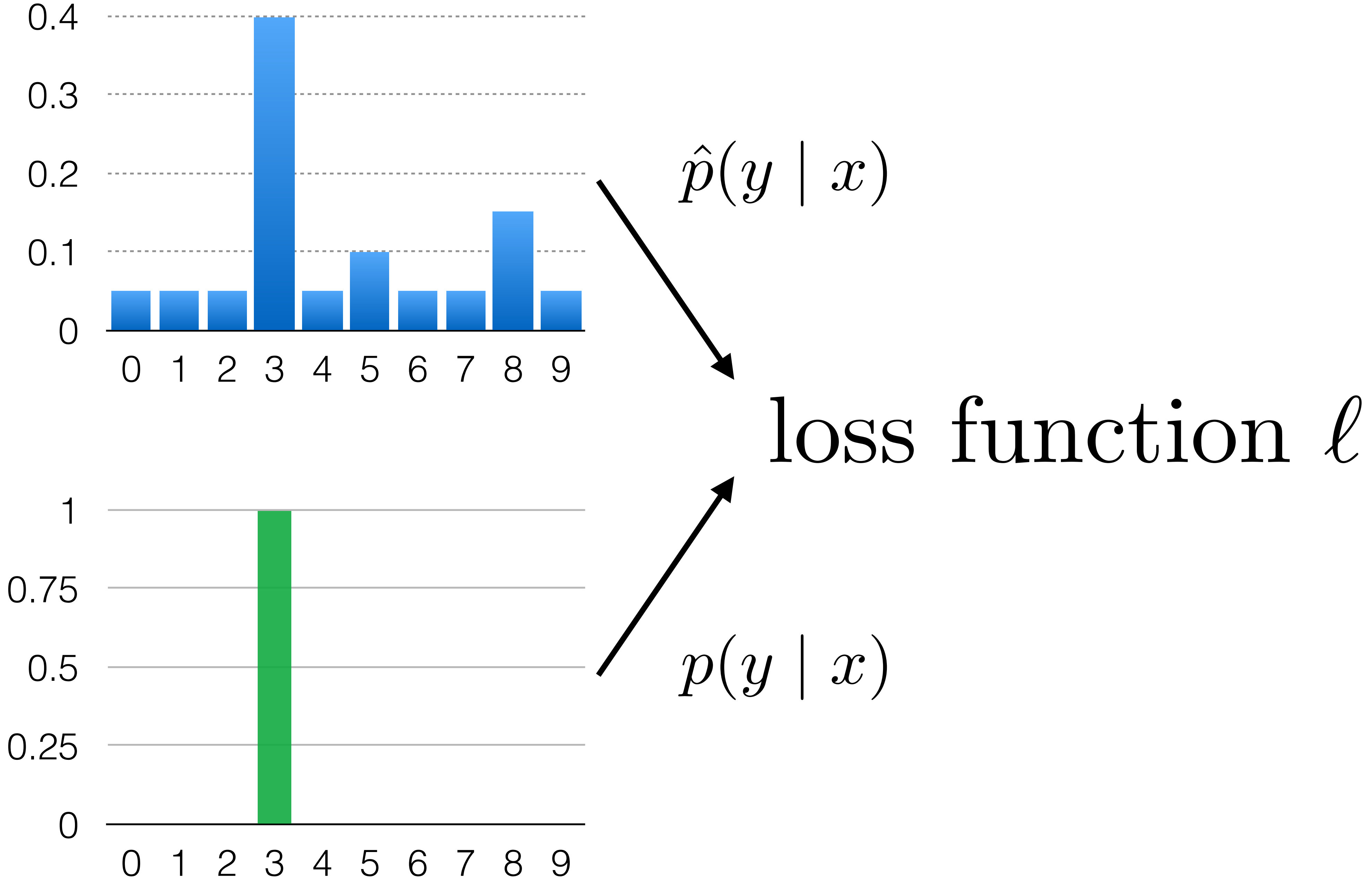
$$\hat{p}(y \mid x)$$

Supervised Learning



$$p(y \mid x)$$

Loss Function

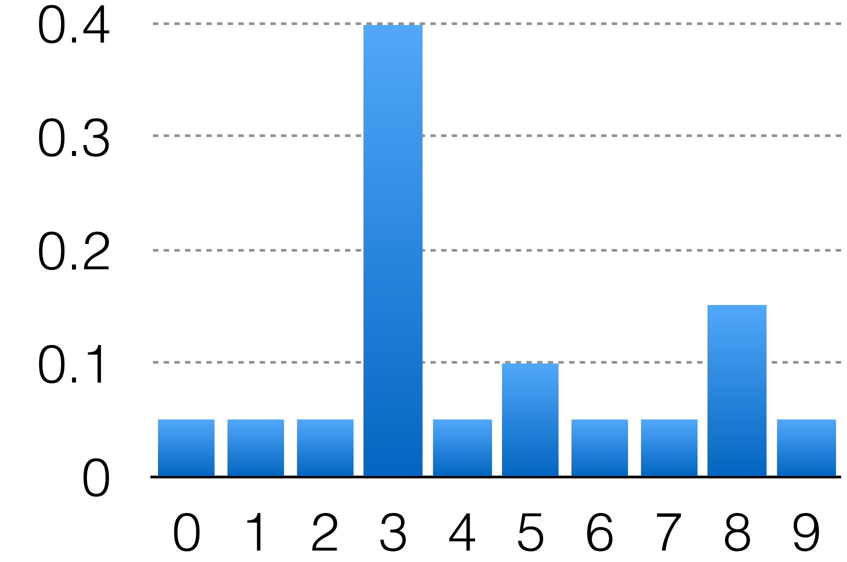


$\hat{p}(y \mid x)$

loss function ℓ

$p(y \mid x)$

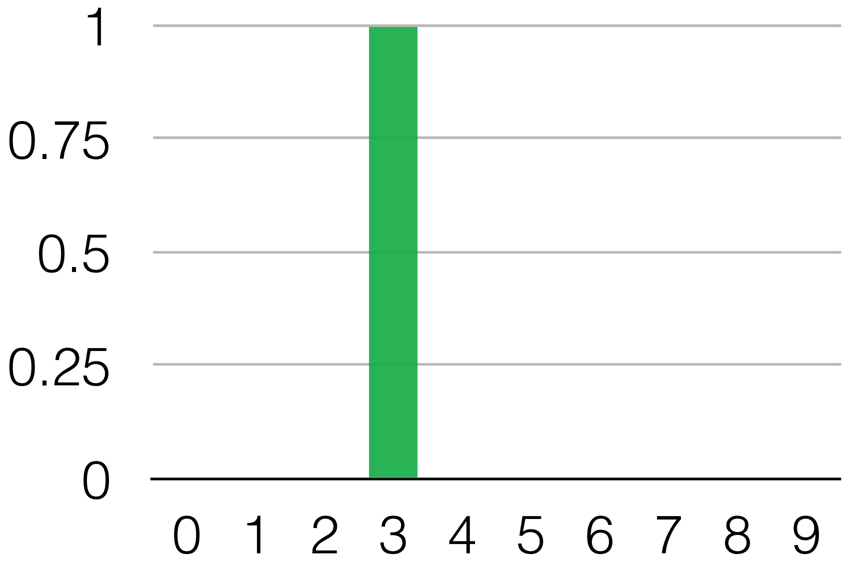
Cross Entropy Loss



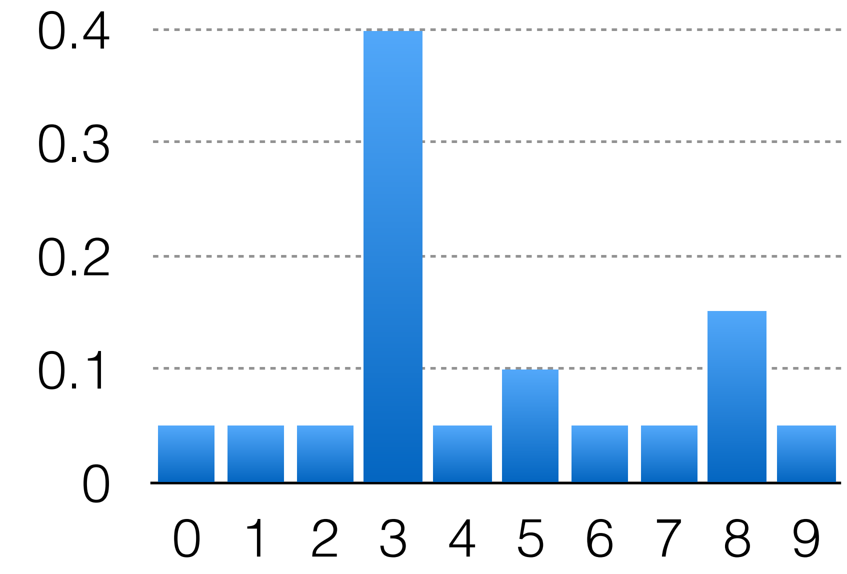
$\hat{p}(y \mid x)$

$$\ell = -\sum_{y \in \{0,1,\ldots,9\}} p(y \mid x) \log \hat{p}(y \mid x)$$

$p(y \mid x)$



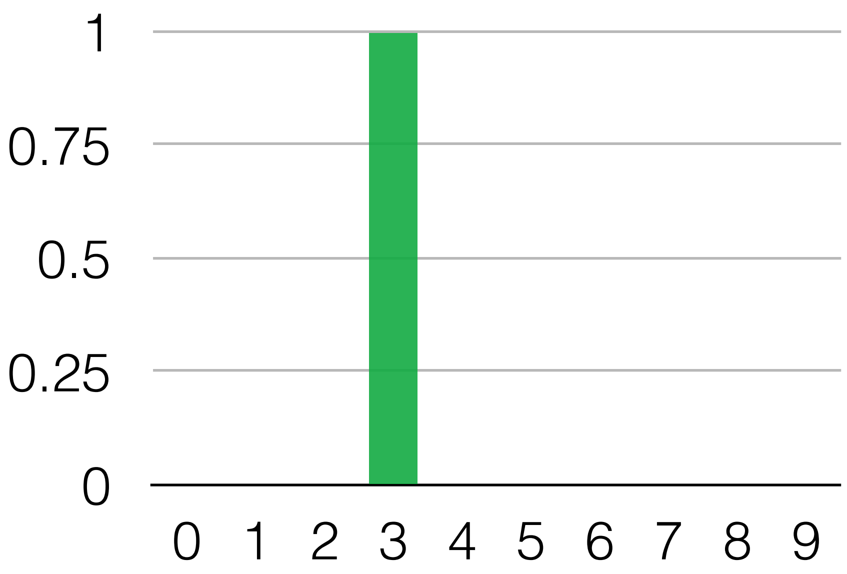
Cross Entropy Loss



$$\hat{p}(y \mid x)$$

$$\ell = -\log \hat{p}(y^* \mid x) \qquad y^* = 3$$

$$p(y \mid x)$$



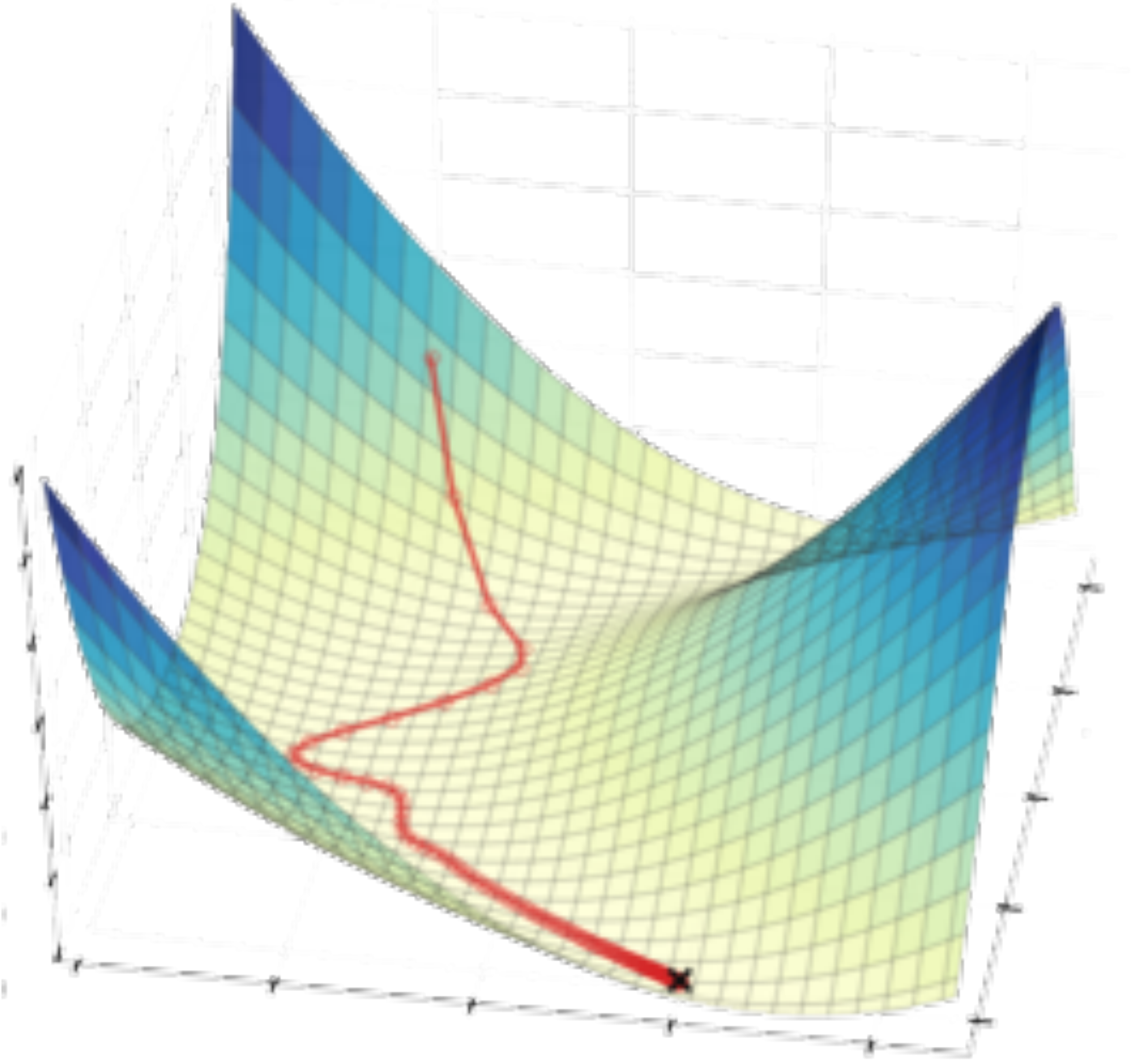
Gradient Descent

$$\ell = -\log \hat{p}_\theta(y^* \mid x)$$

$$\theta \leftarrow \theta - \eta \nabla_\theta \ell$$

$$\ell = -\frac{1}{N} \sum_{n=1}^{N} \log \hat{p}_\theta(y_n^* \mid x_n)$$

$$\ell = -\frac{1}{B} \sum_{n=1}^{B} \log \hat{p}_\theta(y_b^* \mid x_b)$$



Logistic Regression

$$\mathcal{M}$$

$x \in \mathbb{R}^D$

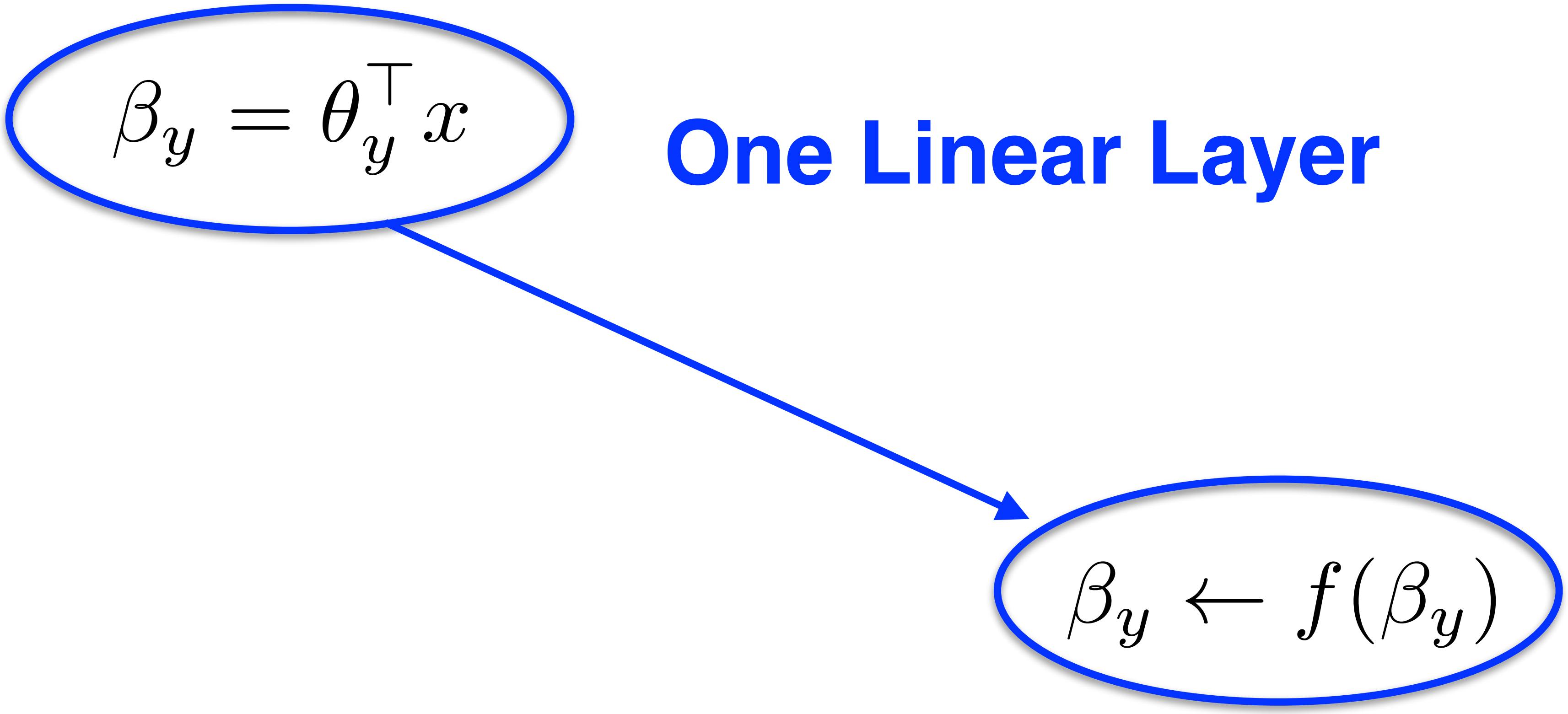
$\beta_y = \theta_y^\top x$

$$\hat{p}_\theta(y \mid x) = \frac{\exp(\beta_y)}{\sum_{y \in \{0,1,\ldots,9\}} \exp(\beta_y)}$$

Softmax Function

Multi-Layer Perceptron

$$\mathcal{M}$$



$$\beta_y = \theta_y^\top x$$

One Linear Layer

$$\beta_y \leftarrow f(\beta_y)$$

One Non-Linear Layer

Multi-Layer Perceptron

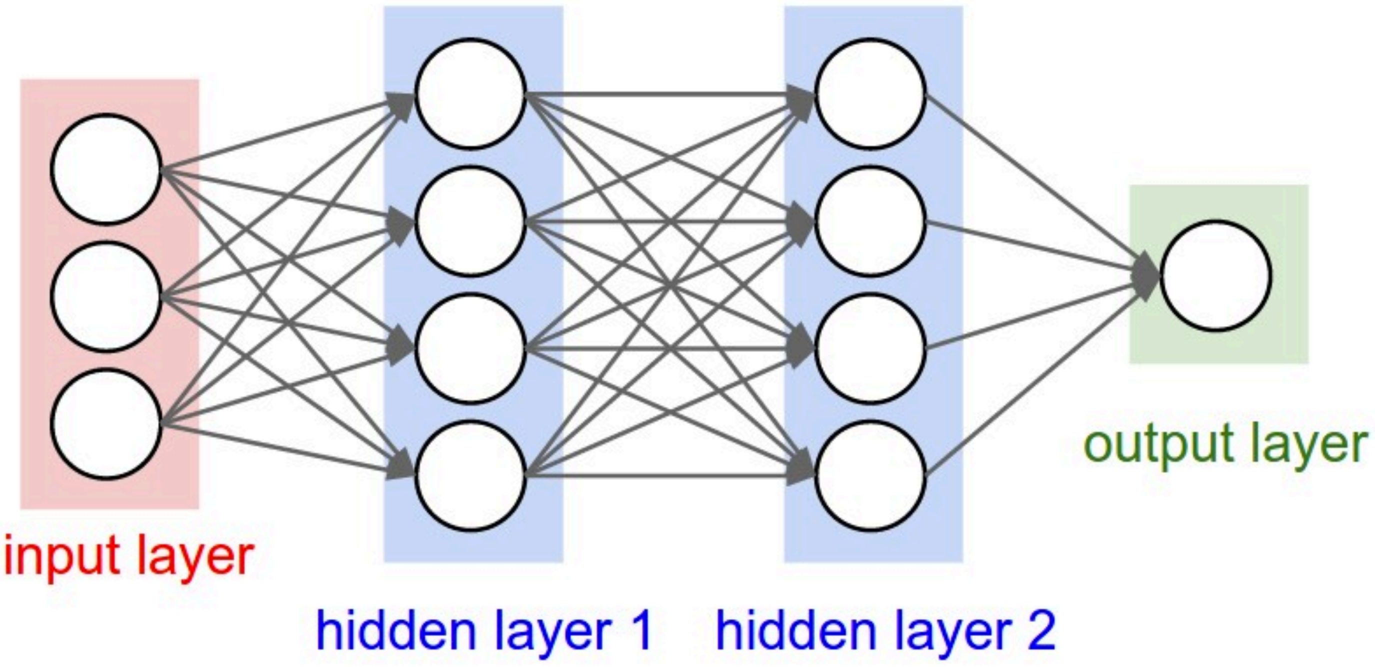
$$\mathcal{M}$$

$$\beta_y = \theta_y^\top x$$

One Linear Layer

$$\beta_y \leftarrow f(\beta_y)$$

One Non-Linear Layer



MXNet Gluon

Let’s make this cool stuff now!

Documents: Dope

Examples: IPython Notebooks